A Largish Sum-of-Squares Implies Circuit Hardness and Derandomization

Pranjal Dutta (CMI & IIT Kanpur)
Nitin Saxena (IIT Kanpur)
Thomas Thierauf (Aalen University)

22^{nd} September, 2020

tMeet @CSE, IIT Madras (Online)

Table of contents

Introduction: Sum-of-squares (SOS)

Sum-of-squares (SOS) Representation

Sum-of-squares (SOS) Representation

An n-variate polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ over a field $\mathbb{F}$ is computed as a *sum-of-squares* (SOS) if

$$f(\boldsymbol{x}) = \sum_{i=1}^{s} c_i \cdot f_i(\boldsymbol{x})^2 \,, \tag{1}$$

for some *top-fanin* s, where $f_i(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$.

Sum-of-squares (SOS) Representation

An n-variate polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ over a field $\mathbb{F}$ is computed as a *sum-of-squares* (SOS) if

$$f(\boldsymbol{x}) = \sum_{i=1}^{s} c_i \cdot f_i(\boldsymbol{x})^2 \,, \tag{1}$$

for some *top-fanin* s, where $f_i(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$.

- Size of f in Eqn. (1) is no. of monomials $= \sum_{i \in [s]} |f_i|_0$.

$|f|_0$ denotes sparsity of f.

Sum-of-squares (SOS) Representation

An n-variate polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ over a field $\mathbb{F}$ is computed as a *sum-of-squares* (SOS) if

$$f(\boldsymbol{x}) = \sum_{i=1}^{s} c_i \cdot f_i(\boldsymbol{x})^2 , \qquad (1)$$

for some *top-fanin* s, where $f_i(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$.

- Size of f in Eqn. (1) is no. of monomials $= \sum_{i \in [s]} |f_i|_0$.

  $|f|_0$ denotes sparsity of f.

  - Eg. $f(x) := 2x + 2 = (x + 3/2)^2 - (x + 1/2)^2$. Size of f in this SOS representation is $2 + 2 = 4$.

Sum-of-squares (SOS) Representation

An n-variate polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ over a field $\mathbb{F}$ is computed as a *sum-of-squares* (SOS) if

$$f(\boldsymbol{x}) = \sum_{i=1}^{s} c_i \cdot f_i(\boldsymbol{x})^2 , \tag{1}$$

for some *top-fanin* s, where $f_i(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$.

- Size of f in Eqn. (1) is no. of monomials $= \sum_{i \in [s]} |f_i|_0$.

  $|f|_0$ denotes sparsity of f.

  - Eg. $f(x) := 2x + 2 = (x + 3/2)^2 - (x + 1/2)^2$. Size of f in this SOS representation is $2 + 2 = 4$.

- Denote the *minimal size* by support-sum $S_{\mathbb{F}}(f)$.

Sum-of-squares (SOS) Representation

An n-variate polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ over a field $\mathbb{F}$ is computed as a *sum-of-squares* (SOS) if

$$f(\boldsymbol{x}) = \sum_{i=1}^{s} c_i \cdot f_i(\boldsymbol{x})^2 \,, \tag{1}$$

for some *top-fanin* s, where $f_i(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$.

- Size of f in Eqn. (1) is no. of monomials $= \sum_{i \in [s]} |f_i|_0$.

  $|f|_0$ denotes sparsity of f.

  - Eg. $f(x) := 2x + 2 = (x+3/2)^2 - (x+1/2)^2$. Size of f in this SOS representation is $2+2=4$.

- Denote the *minimal size* by support-sum $S_{\mathbb{F}}(f)$.

Note. SOS is a *complete* model if $\mathrm{char}(\mathbb{F}) \neq 2$, as $f = \left(\frac{f+1}{2}\right)^2 - \left(\frac{f-1}{2}\right)^2$.

Sum-of-squares (SOS) Representation

An n-variate polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ over a field $\mathbb{F}$ is computed as a *sum-of-squares* (SOS) if

$$f(\boldsymbol{x}) = \sum_{i=1}^{s} c_i \cdot f_i(\boldsymbol{x})^2 , \qquad (1)$$

for some *top-fanin* s, where $f_i(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$.

- Size of f in Eqn. (1) is no. of monomials $=\sum_{i\in[s]} |f_i|_0$.

  $|f|_0$ denotes sparsity of f.

  - Eg. $f(x) := 2x + 2 = (x+3/2)^2 - (x+1/2)^2$. Size of f in this SOS representation is $2+2=4$.

- Denote the *minimal size* by support-sum $S_{\mathbb{F}}(f)$.

Note. SOS is a *complete* model if char$(\mathbb{F}) \neq 2$, as $f = \left(\frac{f+1}{2}\right)^2 - \left(\frac{f-1}{2}\right)^2$.

Trivially, $S_{\mathbb{F}}(f) \leq 2 \cdot (|f|_0 + 1)$, for *any* $f \in \mathbb{F}[\boldsymbol{x}]$.

Upper bound and lower bound: What to expect

- For simplicity, consider univariate SOS representations ($n = 1$).

Upper bound and lower bound: What to expect

- For simplicity, consider univariate SOS representations ($n = 1$).
- For any $\mathrm{char}(\mathbb{F}) \neq 2$ field $\mathbb{F}$:

$$\boxed{|f|_0^{1/2} \;\le\; S_{\mathbb{F}}(f) \;\le\; 2\,|f|_0 + 2}\,. \tag{2}$$

Lower bound by counting monomials:

Upper bound and lower bound: What to expect

- For simplicity, consider univariate SOS representations ($n = 1$).
- For any $\mathrm{char}(\mathbb{F}) \neq 2$ field $\mathbb{F}$:

$$\boxed{|f|_0^{1/2} \;\leq\; S_{\mathbb{F}}(f) \;\leq\; 2\,|f|_0 + 2}\,. \tag{2}$$

Lower bound by counting monomials:

- Suppose $f = \sum_{i=1}^{s} c_i \cdot f_i^2$. Assume, $|f_i|_0 = t_i$.
- Note, $|f_i^2|_0 \leq t_i^2$, for each $i \in [s]$.
- $\sum_{i=1}^{s} t_i^2 \geq |f|_0 \implies \sum_{i=1}^{s} t_i \geq |f|_0^{1/2}$.

Upper bound and lower bound: What to expect

- For simplicity, consider univariate SOS representations ($n = 1$).
- For any $\mathrm{char}(\mathbb{F}) \neq 2$ field $\mathbb{F}$:

$$\boxed{|f|_0^{1/2} \;\leq\; S_{\mathbb{F}}(f) \;\leq\; 2\,|f|_0 + 2}\,. \tag{2}$$

  Lower bound by counting monomials:

  - Suppose $f \;=\; \sum_{i=1}^{s} c_i \cdot f_i^2$. Assume, $|f_i|_0 \;=\; t_i$.
  - Note, $|f_i^2|_0 \;\leq\; t_i^2$, for each $i \in [s]$.
  - $\sum_{i=1}^{s} t_i^2 \;\geq\; |f|_0 \;\implies\; \sum_{i=1}^{s} t_i \;\geq\; |f|_0^{1/2}$.

- If $|f|_0 \approx d$, then $\Omega(d^{1/2}) \;\leq\; S_{\mathbb{F}}(f) \;\leq\; O(d)$.

Upper bound and lower bound: What to expect

- For simplicity, consider univariate SOS representations ($n = 1$).
- For any char($\mathbb{F}$) $\neq 2$ field $\mathbb{F}$:

$$\boxed{|f|_0^{1/2} \leq S_{\mathbb{F}}(f) \leq 2\,|f|_0 + 2}. \tag{2}$$

  Lower bound by counting monomials:
  - Suppose $f = \sum_{i=1}^{s} c_i \cdot f_i^2$. Assume, $|f_i|_0 = t_i$.
  - Note, $|f_i^2|_0 \leq t_i^2$, for each $i \in [s]$.
  - $\sum_{i=1}^{s} t_i^2 \geq |f|_0 \implies \sum_{i=1}^{s} t_i \geq |f|_0^{1/2}$.
- If $|f|_0 \approx d$, then $\Omega(d^{1/2}) \leq S_{\mathbb{F}}(f) \leq O(d)$.
- Does there exist d-degree polynomial $f(x)$ such that $S_{\mathbb{F}}(f) \geq \Omega(d)$?

Upper bound and lower bound: What to expect

- For simplicity, consider univariate SOS representations ($n = 1$).
- For any $\mathrm{char}(\mathbb{F}) \neq 2$ field $\mathbb{F}$:

$$\boxed{|f|_0^{1/2} \ \leq \ S_{\mathbb{F}}(f) \ \leq \ 2\,|f|_0 + 2}. \tag{2}$$

  Lower bound by counting monomials:

  - Suppose $f \ = \ \sum_{i=1}^{s} \ c_i \cdot f_i^2$. Assume, $|f_i|_0 \ = \ t_i$.
  - Note, $|f_i^2|_0 \ \leq \ t_i^2$, for each $i \in [s]$.
  - $\sum_{i=1}^{s} \ t_i^2 \ \geq \ |f|_0 \ \implies \ \sum_{i=1}^{s} \ t_i \ \geq \ |f|_0^{1/2}$.
- If $|f|_0 \approx d$, then $\Omega(d^{1/2}) \ \leq \ S_{\mathbb{F}}(f) \ \leq \ O(d)$.
- Does there exist d-degree polynomial $f(x)$ such that $S_{\mathbb{F}}(f) \geq \Omega(d)$?
  - True for “most" polynomials f, by *dimension-argument*.

Upper bound and lower bound: What to expect

- For simplicity, consider univariate SOS representations ($n = 1$).
- For any $\mathrm{char}(\mathbb{F}) \neq 2$ field $\mathbb{F}$:

$$\boxed{|f|_0^{1/2} \;\leq\; S_{\mathbb{F}}(f) \;\leq\; 2\,|f|_0 + 2}. \tag{2}$$

Lower bound by counting monomials:

  - Suppose $f \;=\; \sum_{i=1}^{s} c_i \cdot f_i^2$. Assume, $|f_i|_0 \;=\; t_i$.
  - Note, $|f_i^2|_0 \;\leq\; t_i^2$, for each $i \in [s]$.
  - $\sum_{i=1}^{s} t_i^2 \;\geq\; |f|_0 \;\implies\; \sum_{i=1}^{s} t_i \;\geq\; |f|_0^{1/2}$.

- If $|f|_0 \approx d$, then $\Omega(d^{1/2}) \;\leq\; S_{\mathbb{F}}(f) \;\leq\; O(d)$.
- Does there exist d-degree polynomial $f(x)$ such that $S_{\mathbb{F}}(f) \geq \Omega(d)$?
  - True for "most" polynomials f, by *dimension-argument*.
  - Assume, $\mathbb{F} = \mathbb{C}$.

Think as *quadratic*-system solving.

Overall Goal

Overall Goal

- **Open Problem.** Find an *explicit* univariate polynomial $f(x) \in \mathbb{C}[x]$ of degree d such that $S(f) \geq \omega(d^{1/2})$.

Overall Goal

- **Open Problem.** Find an *explicit* univariate polynomial $f(x) \in \mathbb{C}[x]$ of degree d such that $S(f) \geq \omega(d^{1/2})$.
  - $S(f) \geq \Omega\left(d/\log d\right)$, where $f(x) = \sum_{i=0}^{d} 2^{2^i} x^i$, using [Strassen'74].

Overall Goal

- **Open Problem.** Find an *explicit* univariate polynomial $f(x) \in \mathbb{C}[x]$ of degree d such that $S(f) \geq \omega(d^{1/2})$.
  - $S(f) \geq \Omega\left(d/\log d\right)$, where $f(x) = \sum_{i=0}^{d} 2^{2^i} x^i$, using [Strassen'74]. But, it is *non-explicit*.

Overall Goal

- **Open Problem.** Find an *explicit* univariate polynomial $f(x) \in \mathbb{C}[x]$ of degree d such that $S(f) \geq \omega(d^{1/2})$.
  - $S(f) \geq \Omega\left(d/\log d\right)$, where $f(x) = \sum_{i=0}^{d} 2^{2^i} x^i$, using [Strassen'74]. But, it is *non-explicit*.
  - To be of any help in complexity theory, polynomials *need* to be explicit. We would work with several definitions of explicitness.
  - Eg. $(x+1)^d$ is 'explicit'.

Overall Goal

- **Open Problem.** Find an *explicit* univariate polynomial $f(x) \in \mathbb{C}[x]$ of degree d such that $S(f) \geq \omega(d^{1/2})$.
  - $S(f) \geq \Omega\,(d/\log d)$, where $f(x) = \sum_{i=0}^{d} 2^{2^i} x^i$, using [Strassen'74]. But, it is *non-explicit*.
  - To be of any help in complexity theory, polynomials *need* to be explicit. We would work with several definitions of explicitness.
  - Eg. $(x+1)^d$ is 'explicit'.
- **Overall Goal (informally):** Show that solving Open Problem implies $\mathsf{VP} \neq \mathsf{VNP}$ (and PIT $\in \mathsf{SUBEXP}$).

SOS Representation – History

SOS Representation – History

- (1770) Lagrange's 4-squares Theorem: Integer as sum of 4-squares.

SOS Representation – History

- (1770) Lagrange's 4-squares Theorem: Integer as sum of 4-squares.
  - Inspired generations of mathematicians [Ramanujan'17].

SOS Representation – History

- (1770) Lagrange's 4-squares Theorem: Integer as sum of 4-squares.
  - Inspired generations of mathematicians [Ramanujan'17].
  - Pythagorean triples, Fermat's 2-squares, Legendre's 3-squares.

SOS Representation – History

- (1770) Lagrange's 4-squares Theorem: Integer as sum of 4-squares.
  - Inspired generations of mathematicians [Ramanujan'17].
  - Pythagorean triples, Fermat's 2-squares, Legendre's 3-squares.
- (1900) Hilbert's *17th problem*: Asks whether a multivariate polynomial, that takes *only* non-negative values over the reals, can be represented as an SOS of rational functions?

SOS Representation – History

- (1770) Lagrange's 4-squares Theorem: Integer as sum of 4-squares.
  - Inspired generations of mathematicians [Ramanujan'17].
  - Pythagorean triples, Fermat's 2-squares, Legendre's 3-squares.
- (1900) Hilbert's *17th problem*: Asks whether a multivariate polynomial, that takes *only* non-negative values over the reals, can be represented as an SOS of rational functions?
  - Note: $c_i = 1$.

SOS Representation – History

- (1770) Lagrange's 4-squares Theorem: Integer as sum of 4-squares.
  - Inspired generations of mathematicians [Ramanujan'17].
  - Pythagorean triples, Fermat's 2-squares, Legendre's 3-squares.
- (1900) Hilbert's *17th problem*: Asks whether a multivariate polynomial, that takes *only* non-negative values over the reals, can be represented as an SOS of rational functions?
  - Note: $c_i = 1$.
- (1990s) SOS constraints appear in convex optimization.

SOS Representation – History

- (1770) Lagrange's 4-squares Theorem: Integer as sum of 4-squares.
  - Inspired generations of mathematicians [Ramanujan'17].
  - Pythagorean triples, Fermat's 2-squares, Legendre's 3-squares.
- (1900) Hilbert's *17th problem*: Asks whether a multivariate polynomial, that takes *only* non-negative values over the reals, can be represented as an SOS of rational functions?
  - Note: $c_i = 1$.
- (1990s) SOS constraints appear in convex optimization.
  - *Lasserre hierarchy* of relaxations in SDP (based on deg).

SOS Representation – History

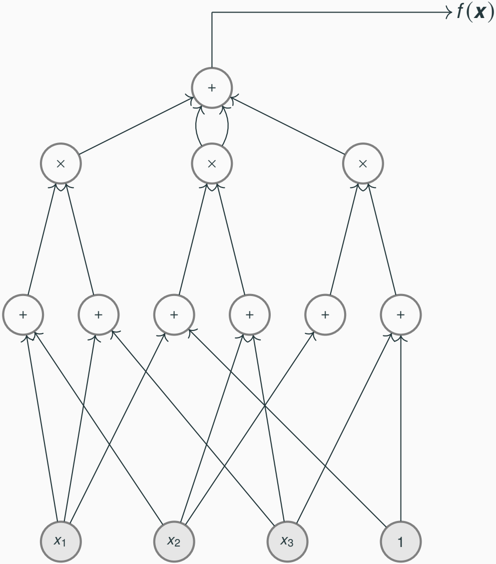
- (1770) Lagrange's 4-squares Theorem: Integer as sum of 4-squares.
  - Inspired generations of mathematicians [Ramanujan'17].
  - Pythagorean triples, Fermat's 2-squares, Legendre's 3-squares.
- (1900) Hilbert's *17th problem*: Asks whether a multivariate polynomial, that takes *only* non-negative values over the reals, can be represented as an SOS of rational functions?
  - Note: $c_i = 1$.
- (1990s) SOS constraints appear in convex optimization.
  - *Lasserre hierarchy* of relaxations in SDP (based on deg).
  - Several applications in approximation, optimization and control theory [Reznick'78, Laurent'09, Barak-Moitra'16].

SOS Representation – History

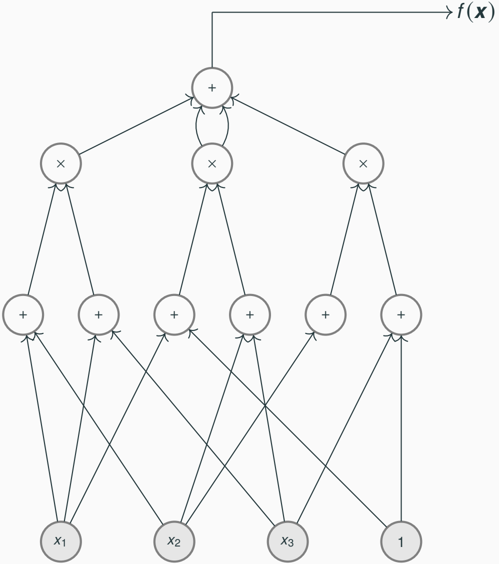
- (1770) Lagrange's 4-squares Theorem: Integer as sum of 4-squares.
  - Inspired generations of mathematicians [Ramanujan'17].
  - Pythagorean triples, Fermat's 2-squares, Legendre's 3-squares.
- (1900) Hilbert's *17th problem*: Asks whether a multivariate polynomial, that takes *only* non-negative values over the reals, can be represented as an SOS of rational functions?
  - Note: $c_i = 1$.
- (1990s) SOS constraints appear in convex optimization.
  - *Lasserre hierarchy* of relaxations in SDP (based on deg).
  - Several applications in approximation, optimization and control theory [Reznick'78, Laurent'09, Barak-Moitra'16].

Basic Algebraic Complexity

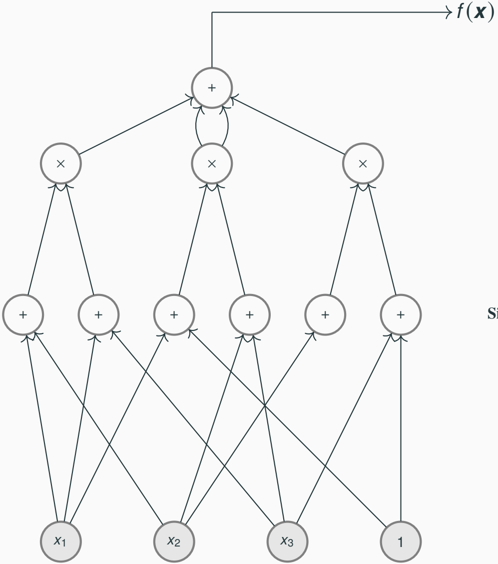
Algebraic Circuits



Algebraic Circuits

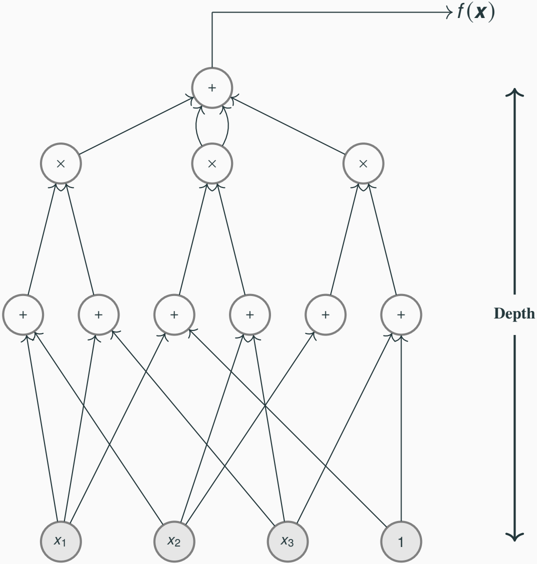


Algebraic Circuits



Size = number of nodes + edges

Algebraic Circuits



- **Valiant's Hypothesis** [Valiant'79]: Symbolic perm_n requires $n^{\omega(1)}$-size circuit.

VP vs. VNP

- **Valiant's Hypothesis** [Valiant'79]: Symbolic perm_n requires $n^{\omega(1)}$-size circuit.
  An *equivalent* statement: Prove $\boxed{\mathsf{VP} \neq \mathsf{VNP}}$.

VP vs. VNP

- **Valiant's Hypothesis** [Valiant'79]: Symbolic perm_n requires $n^{\omega(1)}$-size circuit. An *equivalent* statement: Prove $\boxed{\mathsf{VP} \neq \mathsf{VNP}}$.

- VP : A family $(f_n)_n \in \mathsf{VP}$ (over $\mathbb{F}$) if f_n is a $\mathsf{poly}(n)$-variate polynomial, of degree $\mathsf{poly}(n)$ over $\mathbb{F}$, computed by $\mathsf{poly}(n)$-size circuit.

- **Valiant's Hypothesis** [Valiant'79]: Symbolic perm_n requires $n^{\omega(1)}$-size circuit. An *equivalent* statement: Prove $\boxed{\mathsf{VP} \neq \mathsf{VNP}}$.
- VP : A family $(f_n)_n \in \mathsf{VP}$ (over $\mathbb{F}$) if f_n is a $\mathsf{poly}(n)$-variate polynomial, of degree $\mathsf{poly}(n)$ over $\mathbb{F}$, computed by $\mathsf{poly}(n)$-size circuit.
- VNP : A family $(f_n)_n \in \mathsf{VNP}$ (over $\mathbb{F}$) if $\exists (g_n)_n \in \mathsf{VP}$ & $t(n) = \mathsf{poly}(n)$:

- **Valiant's Hypothesis** [Valiant'79]: Symbolic perm_n requires $n^{\omega(1)}$-size circuit. An *equivalent* statement: Prove $\boxed{\mathsf{VP} \neq \mathsf{VNP}}$.

- VP : A family $(f_n)_n \in \mathsf{VP}$ (over $\mathbb{F}$) if f_n is a $\mathsf{poly}(n)$-variate polynomial, of degree $\mathsf{poly}(n)$ over $\mathbb{F}$, computed by $\mathsf{poly}(n)$-size circuit.

- VNP : A family $(f_n)_n \in \mathsf{VNP}$ (over $\mathbb{F}$) if $\exists (g_n)_n \in \mathsf{VP}$ & $t(n) = \mathsf{poly}(n)$:

$$\boxed{f_n(\boldsymbol{x}) \;=\; \sum_{w \in \{0,1\}^{t(n)}} g_n(\boldsymbol{x}, w)}\;.$$

Polynomial Identity Testing

Polynomial Identity Testing (*PIT*): Given a circuit C, test whether $C \equiv 0$ *(deterministically)*.

Polynomial Identity Testing

Polynomial Identity Testing (*PIT*): Given a circuit C, test whether $C \equiv 0$ *(deterministically)*.

- *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Testing

Polynomial Identity Testing (*PIT*): Given a circuit C, test whether $C \equiv 0$ *(deterministically)*.

- *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma (Ore, Demillo-Lipton, Schwartz, Zippel)

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ is finite, then

$$\text{Prob}_{\boldsymbol{a} \in S^n} \left[P(\boldsymbol{a}) = 0 \right] \leq d/|S| \, .$$

Polynomial Identity Testing

Polynomial Identity Testing (*PIT*): Given a circuit C, test whether $C \equiv 0$ *(deterministically)*.

- *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma (Ore, Demillo-Lipton, Schwartz, Zippel)

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ is finite, then

$$\text{Prob}_{\boldsymbol{a} \in S^n} \left[P(\boldsymbol{a}) = 0 \right] \le d/|S| \ .$$

- The above lemma puts PIT $\in \mathsf{RP}$.

Polynomial Identity Testing

Polynomial Identity Testing (*PIT*): Given a circuit C, test whether $C \equiv 0$ *(deterministically)*.

- *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma (Ore, Demillo-Lipton, Schwartz, Zippel)

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ is finite, then

$$\text{Prob}_{\boldsymbol{a} \in S^n} \left[P(\boldsymbol{a}) = 0 \right] \leq d/|S| \,.$$

- The above lemma puts PIT $\in \mathsf{RP}$.

Hardness-to-randomness (Kabanets-Impagliazzo'04)

$\mathsf{VP} \neq \mathsf{VNP} \implies \mathsf{PIT} \in \mathsf{SUBEXP}$.

Polynomial Identity Testing

Polynomial Identity Testing (*PIT*): Given a circuit C, test whether $C \equiv 0$ *(deterministically)*.

- *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma (Ore, Demillo-Lipton, Schwartz, Zippel)

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ is finite, then

$$\text{Prob}_{\boldsymbol{a} \in S^n} \left[P(\boldsymbol{a}) = 0 \right] \leq d/|S| .$$

- The above lemma puts PIT $\in$ RP.

Hardness-to-randomness (Kabanets-Impagliazzo'04)

$\mathsf{VP} \neq \mathsf{VNP} \implies \text{PIT} \in \mathsf{SUBEXP}$.

- VNP is *exponentially* harder than $\mathsf{VP} \implies$ PIT $\in$ QP.

Polynomial Identity Testing (*PIT*): Given a circuit C, test whether $C \equiv 0$ *(deterministically)*.

- *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma (Ore, Demillo-Lipton, Schwartz, Zippel)

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ is finite, then

$$\text{Prob}_{\boldsymbol{a} \in S^n} \left[P(\boldsymbol{a}) = 0 \right] \leq d/|S| .$$

- The above lemma puts PIT $\in \mathsf{RP}$.

Hardness-to-randomness (Kabanets-Impagliazzo'04)

$\mathsf{VP} \neq \mathsf{VNP} \implies \text{PIT} \in \mathsf{SUBEXP}$.

- VNP is *exponentially* harder than $\mathsf{VP} \implies$ PIT $\in \mathsf{QP}$.
- Efficient PIT $\overset{?}{\implies} \mathsf{VP} \neq \mathsf{VNP}$.

Polynomial Identity Testing (*PIT*): Given a circuit C, test whether $C \equiv 0$ *(deterministically)*.

- *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma (Ore, Demillo-Lipton, Schwartz, Zippel)

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ is finite, then

$$\mathrm{Prob}_{\boldsymbol{a} \in S^n} \left[P(\boldsymbol{a}) = 0 \right] \leq d/|S| \; .$$

- The above lemma puts PIT $\in$ RP.

Hardness-to-randomness (Kabanets-Impagliazzo'04)

$\mathsf{VP} \neq \mathsf{VNP} \implies \mathsf{PIT} \in \mathsf{SUBEXP}$.

- VNP is *exponentially* harder than VP $\implies$ PIT $\in$ QP.
- Efficient PIT $\overset{?}{\implies}$ VP $\neq$ VNP. *Explicitness* is important.

Definition (Explicit Functions). The family $(f_d(x))_d$, where f_d is univariate degree-d polynomial, is *explicit*, if its coefficient-function $\mathrm{coef}_{x_i}(f_d)$ is *easy*:

Explicitness

Definition (Explicit Functions). The family $(f_d(x))_d$, where f_d is univariate degree-d polynomial, is *explicit*, if its coefficient-function $\mathrm{coef}_{x_i}(f_d)$ is *easy*:

- Each coefficient can be at most $\mathsf{poly}(d)$-bits long, and

Explicitness

Definition (Explicit Functions). The family $(f_d(x))_d$, where f_d is univariate degree-d polynomial, is *explicit*, if its coefficient-function $\text{coef}_{x_i}(f_d)$ is *easy*:

- Each coefficient can be at most $\mathsf{poly}(d)$-bits long, and
- the coefficient-function gets input (j, i, d) and outputs the j-th bit of the coefficient of x^i in f_d in

Explicitness

Definition (Explicit Functions). The family $(f_d(x))_d$, where f_d is univariate degree-d polynomial, is *explicit*, if its coefficient-function $\mathrm{coef}_{x_i}(f_d)$ is *easy*:

- Each coefficient can be at most $\mathsf{poly}(d)$-bits long, and
- the coefficient-function gets input (j, i, d) and outputs the j-th bit of the coefficient of x^i in f_d in
  - $\mathsf{poly}(\log d)$-time.

Explicitness

Definition (Explicit Functions). The family $(f_d(x))_d$, where f_d is univariate degree-d polynomial, is *explicit*, if its coefficient-function $\mathrm{coef}_{x_i}(f_d)$ is *easy*:

- Each coefficient can be at most $\mathsf{poly}(d)$-bits long, and
- the coefficient-function gets input (j, i, d) and outputs the j-th bit of the coefficient of x^i in f_d in
  - $\mathsf{poly}(\log d)$-time.
  - Or, ... in $\#\mathsf{P}/\mathsf{poly}$.

Explicitness

Definition (Explicit Functions). The family $(f_d(x))_d$, where f_d is univariate degree-d polynomial, is *explicit*, if its coefficient-function $\mathrm{coef}_{x_i}(f_d)$ is *easy*:

- Each coefficient can be at most $\mathsf{poly}(d)$-bits long, and
- the coefficient-function gets input (j, i, d) and outputs the j-th bit of the coefficient of x^i in f_d in
  - $\mathsf{poly}(\log d)$-time.
  - Or, ... in $\#\mathsf{P}/\mathsf{poly}$.
  - Or, ... in CH.

Explicitness

Definition (Explicit Functions). The family $(f_d(x))_d$, where f_d is univariate degree-d polynomial, is *explicit*, if its coefficient-function $\mathrm{coef}_{x_i}(f_d)$ is *easy*:

- Each coefficient can be at most $\mathsf{poly}(d)$-bits long, and
- the coefficient-function gets input (j, i, d) and outputs the j-th bit of the coefficient of x^i in f_d in
  - $\mathsf{poly}(\log d)$-time.
  - Or, . . . in $\#\mathsf{P}/\mathsf{poly}$.
  - Or, . . . in CH.

Requires GRH to separate VP and VNP.

SOS-hardness

SOS-hardness

Definition (SOS-hardness). An explicit univariate polynomial family $(f_d)_d$ is *SOS-hard*, if $S_{\mathbb{F}}(f_d) = \Omega(d^{0.5+\varepsilon})$, where $\varepsilon := \varepsilon(d) = \omega\left(\sqrt{\frac{\log\log d}{\log d}}\right)$ is a sub-constant function.

SOS-hardness

Definition (SOS-hardness). An explicit univariate polynomial family $(f_d)_d$ is *SOS-hard*, if $S_{\mathbb{F}}(f_d) = \Omega(d^{0.5+\varepsilon})$, where $\varepsilon := \varepsilon(d) = \omega\left(\sqrt{\frac{\log\log d}{\log d}}\right)$ is a sub-constant function.

Remark. Hardness examples– $d^{1/2} \cdot (\log d)^{\sqrt{\log d}}$, $d^{1/2+.01}$.

Definition (SOS-hardness). An explicit univariate polynomial family $(f_d)_d$ is *SOS-hard*, if $S_{\mathbb{F}}(f_d) = \Omega(d^{0.5+\varepsilon})$, where $\varepsilon := \varepsilon(d) = \omega\left(\sqrt{\frac{\log\log d}{\log d}}\right)$ is a sub-constant function.

Remark. Hardness examples– $d^{1/2} \cdot (\log d)^{\sqrt{\log d}}$, $d^{1/2+.01}$.

- There are numerous candidates for $f_d(x)$:

SOS-hardness

Definition (SOS-hardness). An explicit univariate polynomial family $(f_d)_d$ is *SOS-hard*, if $S_{\mathbb{F}}(f_d) = \Omega(d^{0.5+\varepsilon})$, where $\varepsilon := \varepsilon(d) = \omega\left(\sqrt{\frac{\log\log d}{\log d}}\right)$ is a sub-constant function.

Remark. Hardness examples– $d^{1/2} \cdot (\log d)^{\sqrt{\log d}}$, $d^{1/2+.01}$.

- There are numerous candidates for $f_d(x)$:
  - The famous *Pochhammer-Wilkinson* polynomial $f_d := \prod_{i=1}^{d}(x-i)$.

SOS-hardness

Definition (SOS-hardness). An explicit univariate polynomial family $(f_d)_d$ is *SOS-hard*, if $S_{\mathbb{F}}(f_d) = \Omega(d^{0.5+\varepsilon})$, where $\varepsilon := \varepsilon(d) = \omega\left(\sqrt{\frac{\log\log d}{\log d}}\right)$ is a sub-constant function.

Remark. Hardness examples– $d^{1/2} \cdot (\log d)^{\sqrt{\log d}}$, $d^{\,1/2+.01}$.

- There are numerous candidates for $f_d(x)$:
  - The famous *Pochhammer-Wilkinson* polynomial $f_d := \prod_{i=1}^{d}(x-i)$.
  - $f_d := \sum_{i=0}^{d} 2^{i^2} x^i$.

SOS-hardness

Definition (SOS-hardness). An explicit univariate polynomial family $(f_d)_d$ is *SOS-hard*, if $S_{\mathbb{F}}(f_d) = \Omega(d^{0.5+\varepsilon})$, where $\varepsilon := \varepsilon(d) = \omega\left(\sqrt{\frac{\log\log d}{\log d}}\right)$ is a sub-constant function.

Remark. Hardness examples– $d^{1/2} \cdot (\log d)^{\sqrt{\log d}}$, $d^{1/2+.01}$.

- There are numerous candidates for $f_d(x)$:
  - The famous *Pochhammer-Wilkinson* polynomial $f_d := \prod_{i=1}^{d}(x-i)$.
  - $f_d := \sum_{i=0}^{d} 2^{i^2} x^i$.

$\sum_{i=0}^{d} 2^i x^i$ is *not* a candidate

Definition (SOS-hardness). An explicit univariate polynomial family $(f_d)_d$ is *SOS-hard*, if $S_{\mathbb{F}}(f_d) = \Omega(d^{0.5+\varepsilon})$, where $\varepsilon := \varepsilon(d) = \omega\left(\sqrt{\frac{\log\log d}{\log d}}\right)$ is a sub-constant function.

Remark. Hardness examples– $d^{1/2} \cdot (\log d)^{\sqrt{\log d}}$, $d^{1/2+.01}$.

- There are numerous candidates for $f_d(x)$:
  - The famous *Pochhammer-Wilkinson* polynomial $f_d := \prod_{i=1}^{d}(x-i)$.
  - $f_d := \sum_{i=0}^{d} 2^{i^2} x^i$.
  - $f_d := (x+1)^d$.

$\sum_{i=0}^{d} 2^i x^i$ is *not* a candidate

SOS-hardness

Definition (SOS-hardness). An explicit univariate polynomial family $(f_d)_d$ is *SOS-hard*, if $S_{\mathbb{F}}(f_d) = \Omega(d^{0.5+\varepsilon})$, where $\varepsilon := \varepsilon(d) = \omega\left(\sqrt{\frac{\log\log d}{\log d}}\right)$ is a sub-constant function.

Remark. Hardness examples– $d^{1/2} \cdot (\log d)^{\sqrt{\log d}}$, $d^{\,1/2+.01}$.

- There are numerous candidates for $f_d(x)$:
  - The famous *Pochhammer-Wilkinson* polynomial $f_d := \prod_{i=1}^{d}(x-i)$.
  - $f_d := \sum_{i=0}^{d} 2^{i^2} x^i$.
  - $f_d := (x+1)^d$.

$\sum_{i=0}^{d} 2^i x^i$ is *not* a candidate

$(x+1)^d$ has $\mathrm{poly}(\log d)$-size circuit.

SOS-hardness and comparison with prior works

SOS-hardness is quite *incomparable/weak* to previous works:

SOS-hardness and comparison with prior works

SOS-hardness is quite *incomparable/weak* to previous works:

- [Agrawal-Vinay'08,..,Gupta-Kamath-Kayal-Saptharishi'13,..,Agrawal-Ghosh-Saxena'18] Hardness for special depth-4/3 – sum-of *unbounded-powers* of *multivariates* $\sum \bigwedge^{\omega(1)} \sum \prod$.

SOS-hardness and comparison with prior works

SOS-hardness is quite *incomparable/weak* to previous works:

- [Agrawal-Vinay'08,..,Gupta-Kamath-Kayal-Saptharishi'13,..,Agrawal-Ghosh-Saxena'18] Hardness for special depth-4/3 – sum-of *unbounded-powers* of *multivariates* $\sum \bigwedge^{\omega(1)} \sum \prod$.
- [koiran'11] Used univariate depth-4 expression of *unbounded-powers*; also lower bound on the *top-fanin* (we require SOS-size).

SOS-hardness and comparison with prior works

SOS-hardness is quite *incomparable/weak* to previous works:

- [Agrawal-Vinay'08,..,Gupta-Kamath-Kayal-Saptharishi'13,..,Agrawal-Ghosh-Saxena'18] Hardness for special depth-4/3 – sum-of *unbounded-powers* of *multivariates* $\sum \bigwedge^{\omega(1)} \sum \prod$.
- [koiran'11] Used univariate depth-4 expression of *unbounded-powers*; also lower bound on the *top-fanin* (we require SOS-size).
  - SOS-size is *neither* top-fanin nor the "size" of the depth-4 circuits, rather it is # $\prod$-operations in $\sum \bigwedge^2 \sum \prod$-formula.

SOS-hardness and comparison with prior works

SOS-hardness is quite *incomparable/weak* to previous works:

- [Agrawal-Vinay'08,..,Gupta-Kamath-Kayal-Saptharishi'13,..,Agrawal-Ghosh-Saxena'18] Hardness for special depth-4/3 – sum-of *unbounded-powers* of *multivariates* $\sum \bigwedge^{\omega(1)} \sum \prod$.
- [koiran'11] Used univariate depth-4 expression of *unbounded-powers*; also lower bound on the *top-fanin* (we require SOS-size).
  - SOS-size is *neither* top-fanin nor the "size" of the depth-4 circuits, rather it is # $\prod$-operations in $\sum \bigwedge^2 \sum \prod$-formula.
  - Circuit-hardness $\implies$ SOS-hardness (f requires s size circuit implies $S(f) \geq s/\log d$); the **opposite** plausibly *doesn't* hold.

SOS-hardness and comparison with prior works

SOS-hardness is quite *incomparable/weak* to previous works:

- [Agrawal-Vinay'08,..,Gupta-Kamath-Kayal-Saptharishi'13,..,Agrawal-Ghosh-Saxena'18] Hardness for special depth-4/3 – sum-of *unbounded-powers* of *multivariates* $\sum \wedge^{\omega(1)} \sum \prod$.
- [koiran'11] Used univariate depth-4 expression of *unbounded-powers*; also lower bound on the *top-fanin* (we require SOS-size).
  - SOS-size is *neither* top-fanin nor the "size" of the depth-4 circuits, rather it is # $\prod$-operations in $\sum \wedge^2 \sum \prod$-formula.
  - Circuit-hardness $\implies$ SOS-hardness (f requires s size circuit implies $S(f) \geq s/\log d$); the **opposite** plausibly *doesn't* hold.
- real-τ-conjecture [Koiran'10] and [Koiran-Portier-Tavenas-Thomassé'15] Newton-polygon-τ-conjecture about roots of similar depth-4 expressions (also here, $\omega(\sqrt{d})$ vs. d).

SOS-hardness and comparison with prior works

SOS-hardness is quite *incomparable/weak* to previous works:

- [Agrawal-Vinay'08,..,Gupta-Kamath-Kayal-Saptharishi'13,..,Agrawal-Ghosh-Saxena'18] Hardness for special depth-4/3 – sum-of *unbounded-powers* of *multivariates* $\sum \bigwedge^{\omega(1)} \sum \prod$.
- [koiran'11] Used univariate depth-4 expression of *unbounded-powers*; also lower bound on the *top-fanin* (we require SOS-size).
  - SOS-size is *neither* top-fanin nor the "size" of the depth-4 circuits, rather it is # $\prod$-operations in $\sum \bigwedge^2 \sum \prod$-formula.
  - Circuit-hardness $\implies$ SOS-hardness (f requires s size circuit implies $S(f) \geq s/\log d$); the **opposite** plausibly *doesn't* hold.
- real-τ-conjecture [Koiran'10] and [Koiran-Portier-Tavenas-Thomassé'15] Newton-polygon-τ-conjecture about roots of similar depth-4 expressions (also here, $\omega(\sqrt{d})$ vs. d).
- [Raz'08] Super-poly-elusive functions eluding degree-2 maps (generic *multivariate*).

SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Theorem 1 (Dutta-Saxena-Thierauf'20)

If there exists an SOS-hard polynomial family, then $\mathsf{VP} \neq \mathsf{VNP}$.

SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Theorem 1 (Dutta-Saxena-Thierauf'20)
If there exists an SOS-hard polynomial family, then $\mathsf{VP} \neq \mathsf{VNP}$.

- Natural analogue of SOS lower bound to hardness of Permanent in the *non-commutative* settings, [Hrubeš-Wigderson-Yehudayoff'11].

Theorem 1 (Dutta-Saxena-Thierauf'20)
If there exists an SOS-hard polynomial family, then VP $\neq$ VNP.

- Natural analogue of SOS lower bound to hardness of Permanent in the *non-commutative* settings, [Hrubeš-Wigderson-Yehudayoff'11].
- Restrict the degrees of f_i to be $d \cdot o(\log d)$ and the top-fanin $s = d^{o(1)}$.

SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Theorem 1 (Dutta-Saxena-Thierauf'20)
If there exists an SOS-hard polynomial family, then $\mathsf{VP} \neq \mathsf{VNP}$.

- Natural analogue of SOS lower bound to hardness of Permanent in the *non-commutative* settings, [Hrubeš-Wigderson-Yehudayoff'11].
- Restrict the degrees of f_i to be $d \cdot o(\log d)$ and the top-fanin $s = d^{o(1)}$.
- A stronger SOS-hardness notion with *constant* ε, gives an *exponential* separation between VP and VNP. This proof has many technical differences.

SOS-hardness and VP vs. VNP

Main Lemma (SOS Decomposition)

Main Lemma (SOS Decomposition)

Let $\mathbb{F}$ be a field of characteristic $\neq 2$. Let $f(\boldsymbol{x})$ be an n-variate polynomial over $\mathbb{F}$ of degree d, computed by a circuit of size s. Then there exist $f_i \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$ such that

$$f(\boldsymbol{x}) = \sum_{i=1}^{s'} c_i f_i(\boldsymbol{x})^2 ,$$

where $s' \leq (sd)^{O(\log d)}$, and $\deg(f_i) \leq \lceil d/2 \rceil$, for all $i \in [s']$.

SOS Decomposition

Main Lemma (SOS Decomposition)

Let $\mathbb{F}$ be a field of characteristic $\neq 2$. Let $f(\boldsymbol{x})$ be an n-variate polynomial over $\mathbb{F}$ of degree d, computed by a circuit of size s. Then there exist $f_i \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$ such that

$$f(\boldsymbol{x}) = \sum_{i=1}^{s'} c_i f_i(\boldsymbol{x})^2 ,$$

where $s' \leq (sd)^{O(\log d)}$, and $\deg(f_i) \leq \lceil d/2 \rceil$, for all $i \in [s']$.

Can we *improve* s' to $\mathrm{poly}(sd)$?

Algebraic branching programs (ABP). An ABP is a directed acyclic graph with a *starting vertex* s with in-degree zero, an *end vertex* t with out-degree zero. The edge labels are $a_1x_1 + \ldots + a_nx_n + c \in \mathbb{F}[\boldsymbol{x}]$, where $a_i, c \in \mathbb{F}$.

Algebraic branching programs (ABP). An ABP is a directed acyclic graph with a *starting vertex* s with in-degree zero, an *end vertex* t with out-degree zero. The edge labels are $a_1x_1 + \ldots + a_nx_n + c \in \mathbb{F}[\boldsymbol{x}]$, where $a_i, c \in \mathbb{F}$.

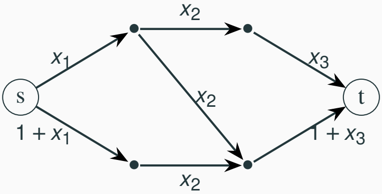
- The *weight of a path* is the product of labels of the edges in the path.

Algebraic branching programs (ABP). An ABP is a directed acyclic graph with a *starting vertex* s with in-degree zero, an *end vertex* t with out-degree zero. The edge labels are $a_1x_1 + \ldots + a_nx_n + c \in \mathbb{F}[\boldsymbol{x}]$, where $a_i, c \in \mathbb{F}$.

- The *weight of a path* is the product of labels of the edges in the path.
- The *polynomial computed by the ABP* is the polynomial computed at the end vertex t.

Algebraic branching programs (ABP). An ABP is a directed acyclic graph with a *starting vertex* s with in-degree zero, an *end vertex* t with out-degree zero. The edge labels are $a_1x_1 + \ldots + a_nx_n + c \in \mathbb{F}[\boldsymbol{x}]$, where $a_i, c \in \mathbb{F}$.

- The *weight of a path* is the product of labels of the edges in the path.
- The *polynomial computed by the ABP* is the polynomial computed at the end vertex t.



This ABP computes

$$x_1x_2x_3 + x_1x_2(1+x_3) + (1+x_1)x_2(1+x_3)$$

Proof idea of Main Lemma

Proof Sketch. Here is the basic outline:

Proof idea of Main Lemma

Proof Sketch. Here is the basic outline:

- Wlog, assume it to be a homogeneous f of degree d computed by size s circuit.

Proof idea of Main Lemma

Proof Sketch. Here is the basic outline:

- Wlog, assume it to be a homogeneous f of degree d computed by size s circuit.
- Apply result of [Valiant-Skyum-Berkowitz-Rackoff'83] to make it $\log$-depth with $\mathsf{poly}(s)$-size blowup.

Proof idea of Main Lemma

Proof Sketch. Here is the basic outline:

- Wlog, assume it to be a homogeneous f of degree d computed by size s circuit.
- Apply result of [Valiant-Skyum-Berkowitz-Rackoff'83] to make it $\log$-depth with $\mathsf{poly}(s)$-size blowup.
- Convert the circuit to a *homogeneous* ABP of size (width) $w := s^{\log d}$ such that *each* edge has *linear form* weight (without constants).

Proof idea of Main Lemma

Proof Sketch. Here is the basic outline:

- Wlog, assume it to be a homogeneous f of degree d computed by size s circuit.
- Apply result of [Valiant-Skyum-Berkowitz-Rackoff'83] to make it $\log$-depth with $\mathsf{poly}(s)$-size blowup.
- Convert the circuit to a *homogeneous* ABP of size (width) $w := s^{\log d}$ such that *each* edge has *linear form* weight (without constants).
- By construction, i-th layer nodes compute polynomials of degree *exactly* i.

Proof Sketch. Here is the basic outline:

- Wlog, assume it to be a homogeneous f of degree d computed by size s circuit.
- Apply result of [Valiant-Skyum-Berkowitz-Rackoff'83] to make it $\log$-depth with $\mathsf{poly}(s)$-size blowup.
- Convert the circuit to a *homogeneous* ABP of size (width) $w := s^{\log d}$ such that *each* edge has *linear form* weight (without constants).
- By construction, i-th layer nodes compute polynomials of degree *exactly* i.
- *Cut* the ABP, at the $d/2$-th layer, we get $f = (f_1, \ldots, f_w)^T \cdot \left(f_1', \ldots, f_w'\right) = \sum_{i=1}^{w} f_i \cdot f_i'$, where f_i and f_i' have degree $d/2$.

Proof idea of Main Lemma

Proof Sketch. Here is the basic outline:

- Wlog, assume it to be a homogeneous f of degree d computed by size s circuit.
- Apply result of [Valiant-Skyum-Berkowitz-Rackoff'83] to make it $\log$-depth with $\mathsf{poly}(s)$-size blowup.
- Convert the circuit to a *homogeneous* ABP of size (width) $w := s^{\log d}$ such that *each* edge has *linear form* weight (without constants).
- By construction, i-th layer nodes compute polynomials of degree *exactly* i.
- *Cut* the ABP, at the $d/2$-th layer, we get $f \ = \ (f_1, \ldots, f_w)^T \cdot \left(f'_1, \ldots, f'_w\right) \ = \ \sum_{i=1}^{w} f_i \cdot f'_i$, where f_i and f'_i have degree $d/2$.
- Write each product $f_i \cdot f'_i \ = \ 1/4 \cdot (f_i + f'_i)^2 \ - \ 1/4 \cdot (f_i - f'_i)^2$, which finally gives the desired decomposition.

Proof of Theorem 1: SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Recall Theorem 1: If an explicit $f_d(x)$ is SOS-hard i.e. $S_{\mathbb{F}}(f_d) \geq d^{1/2+\varepsilon}$ for $\varepsilon = \omega(\sqrt{\log\log d/\log d})$, then $\mathsf{VP} \neq \mathsf{VNP}$.

Proof of Theorem 1: SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Recall Theorem 1: If an explicit $f_d(x)$ is SOS-hard i.e. $S_{\mathbb{F}}(f_d) \geq d^{1/2+\varepsilon}$ for $\varepsilon = \omega(\sqrt{\log\log d/\log d})$, then $\mathsf{VP} \neq \mathsf{VNP}$.

- ❑ Wlog, f_d is SOS-hard with $\varepsilon = (\log\log d/\log d)^{1/3}$.

Proof of Theorem 1: SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Recall Theorem 1: If an explicit $f_d(x)$ is SOS-hard i.e. $S_{\mathbb{F}}(f_d) \geq d^{1/2+\varepsilon}$ for $\varepsilon = \omega(\sqrt{\log\log d/\log d})$, then $\mathsf{VP} \neq \mathsf{VNP}$.

- ❑ Wlog, f_d is SOS-hard with $\varepsilon = (\log\log d/\log d)^{1/3}$.
- ❑ Convert this to a kn-variate n-degree multilinear polynomial $P_{n,k}$ where $k^n \geq d > (k-1)^n$, (n and k are both functions of d to be fixed later) and show that the family $\in \mathsf{VNP}$, but $\notin \mathsf{VP}$.

Proof of Theorem 1: SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Recall Theorem 1: If an explicit $f_d(x)$ is SOS-hard i.e. $S_{\mathbb{F}}(f_d) \geq d^{1/2+\varepsilon}$ for $\varepsilon = \omega(\sqrt{\log\log d/\log d})$, then $\mathsf{VP} \neq \mathsf{VNP}$.

- Wlog, f_d is SOS-hard with $\varepsilon = (\log\log d/\log d)^{1/3}$.
- Convert this to a kn-variate n-degree multilinear polynomial $P_{n,k}$ where $k^n \geq d > (k-1)^n$, (n and k are both functions of d to be fixed later) and show that the family $\in \mathsf{VNP}$, but $\notin \mathsf{VP}$. The conversion is as follows:

Proof of Theorem 1: SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Recall Theorem 1: If an explicit $f_d(x)$ is SOS-hard i.e. $S_{\mathbb{F}}(f_d) \geq d^{1/2+\varepsilon}$ for $\varepsilon = \omega(\sqrt{\log\log d/\log d})$, then $\mathsf{VP} \neq \mathsf{VNP}$.

- ❑ Wlog, f_d is SOS-hard with $\varepsilon = (\log\log d/\log d)^{1/3}$.
- ❑ Convert this to a kn-variate n-degree multilinear polynomial $P_{n,k}$ where $k^n \geq d > (k-1)^n$, (n and k are both functions of d to be fixed later) and show that the family $\in \mathsf{VNP}$, but $\notin \mathsf{VP}$. The conversion is as follows:
  - ➢ Introduce new variables $y_{j,\ell}$ where $j \in [n]$ and $\ell \in [0, k-1]$.

Proof of Theorem 1: SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Recall Theorem 1: If an explicit $f_d(x)$ is SOS-hard i.e. $S_{\mathbb{F}}(f_d) \geq d^{1/2+\varepsilon}$ for $\varepsilon = \omega(\sqrt{\log\log d/\log d})$, then $\mathsf{VP} \neq \mathsf{VNP}$.

- Wlog, f_d is SOS-hard with $\varepsilon = (\log\log d/\log d)^{1/3}$.
- Convert this to a kn-variate n-degree multilinear polynomial $P_{n,k}$ where $k^n \geq d > (k-1)^n$, (n and k are both functions of d to be fixed later) and show that the family $\in \mathsf{VNP}$, but $\notin \mathsf{VP}$. The conversion is as follows:
  - Introduce new variables $y_{j,\ell}$ where $j \in [n]$ and $\ell \in [0, k-1]$.
  - Monomial x^i in $f_d(x)$ maps to $\phi(x^i) := \prod_{j=1}^{n} y_{j,i_j}$, where $i =: \sum_{j=1}^{n} i_j \cdot k^{j-1}, \ 0 \leq i_j \leq k-1$.

Proof of Theorem 1: SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Recall Theorem 1: If an explicit $f_d(x)$ is SOS-hard i.e. $S_{\mathbb{F}}(f_d) \geq d^{1/2+\varepsilon}$ for $\varepsilon = \omega(\sqrt{\log\log d/\log d})$, then $\mathsf{VP} \neq \mathsf{VNP}$.

- Wlog, f_d is SOS-hard with $\varepsilon = (\log\log d/\log d)^{1/3}$.
- Convert this to a kn-variate n-degree multilinear polynomial $P_{n,k}$ where $k^n \geq d > (k-1)^n$, (n and k are both functions of d to be fixed later) and show that the family $\in \mathsf{VNP}$, but $\notin \mathsf{VP}$. The conversion is as follows:
  - Introduce new variables $y_{j,\ell}$ where $j \in [n]$ and $\ell \in [0, k-1]$.
  - Monomial x^i in $f_d(x)$ maps to $\phi(x^i) := \prod_{j=1}^{n} y_{j,i_j}$, where $i =: \sum_{j=1}^{n} i_j \cdot k^{j-1}, \ 0 \leq i_j \leq k-1$.
  - By definition $P_{n,k} = \phi(f_d)$ is kn-variate n-degree multilinear polynomial.

Proof of Theorem 1: SOS-hardness to $\mathsf{VP} \neq \mathsf{VNP}$

Recall Theorem 1: If an explicit $f_d(x)$ is SOS-hard i.e. $S_{\mathbb{F}}(f_d) \geq d^{1/2+\varepsilon}$ for $\varepsilon = \omega(\sqrt{\log\log d/\log d})$, then $\mathsf{VP} \neq \mathsf{VNP}$.

- Wlog, f_d is SOS-hard with $\varepsilon = (\log\log d/\log d)^{1/3}$.
- Convert this to a kn-variate n-degree multilinear polynomial $P_{n,k}$ where $k^n \geq d > (k-1)^n$, (n and k are both functions of d to be fixed later) and show that the family $\in \mathsf{VNP}$, but $\notin \mathsf{VP}$. The conversion is as follows:
  - Introduce new variables $y_{j,\ell}$ where $j \in [n]$ and $\ell \in [0, k-1]$.
  - Monomial x^i in $f_d(x)$ maps to $\phi(x^i) := \prod_{j=1}^{n} y_{j,i_j}$, where $i =: \sum_{j=1}^{n} i_j \cdot k^{j-1},\ \ 0 \leq i_j \leq k-1$.
  - By definition $P_{n,k} = \phi(f_d)$ is kn-variate n-degree multilinear polynomial.
- $P_{n,k}$ is very explicit and thus the family $\in \mathsf{VNP}$.

Proof of Theorem 1 (continued)

Proof of Theorem 1 (continued)

- We show that circuit-size($P_{n,k}$) = $(kn)^{\omega(1)}$ (implying the family $\notin$ VP).

Proof of Theorem 1 (continued)

- We show that circuit-size($P_{n,k}$) = $(kn)^{\omega(1)}$ (implying the family $\notin \mathsf{VP}$).
- Proof by contradiction. Suppose $P_{n,k}$ has a small-size circuit.

Proof of Theorem 1 (continued)

- We show that circuit-size$(P_{n,k}) = (kn)^{\omega(1)}$ (implying the family $\notin \mathsf{VP}$).
- Proof by contradiction. Suppose $P_{n,k}$ has a small-size circuit.
- SOS Decomposition shows that $P_{n,k}(\boldsymbol{y}) = \sum_{i=1}^{s'} c_i \cdot Q_i(\boldsymbol{y})^2$, where $\deg(Q_i) \leq \deg(P_{n,k})/2 \leq n/2$.

Proof of Theorem 1 (continued)

- We show that circuit-size$(P_{n,k}) = (kn)^{\omega(1)}$ (implying the family $\notin \mathsf{VP}$).
- Proof by contradiction. Suppose $P_{n,k}$ has a small-size circuit.
- SOS Decomposition shows that $P_{n,k}(\boldsymbol{y}) = \sum_{i=1}^{s'} c_i \cdot Q_i(\boldsymbol{y})^2$, where $\deg(Q_i) \le \deg(P_{n,k})/2 \le n/2$.
- Apply ϕ both side to get $f_d = \phi(P_{n,k}) = \sum_{i=1}^{s'} c_i \cdot \phi(Q_i)^2$.

Proof of Theorem 1 (continued)

- We show that circuit-size$(P_{n,k}) = (kn)^{\omega(1)}$ (implying the family $\notin \mathsf{VP}$).
- Proof by contradiction. Suppose $P_{n,k}$ has a small-size circuit.
- SOS Decomposition shows that $P_{n,k}(\boldsymbol{y}) = \sum_{i=1}^{s'} c_i \cdot Q_i(\boldsymbol{y})^2$, where $\deg(Q_i) \le \deg(P_{n,k})/2 \le n/2$.
- Apply ϕ both side to get $f_d = \phi(P_{n,k}) = \sum_{i=1}^{s'} c_i \cdot \phi(Q_i)^2$.
- ϕ *cannot* increase the sparsity. Thus, $|\phi(Q_i)|_0 \le |Q_i|_0 \le \binom{kn+n/2}{n/2}$.

Proof of Theorem 1 (continued)

- We show that circuit-size$(P_{n,k}) = (kn)^{\omega(1)}$ (implying the family $\notin \mathsf{VP}$).
- Proof by contradiction. Suppose $P_{n,k}$ has a small-size circuit.
- SOS Decomposition shows that $P_{n,k}(\boldsymbol{y}) = \sum_{i=1}^{s'} c_i \cdot Q_i(\boldsymbol{y})^2$, where $\deg(Q_i) \le \deg(P_{n,k})/2 \le n/2$.
- Apply ϕ both side to get $f_d = \phi(P_{n,k}) = \sum_{i=1}^{s'} c_i \cdot \phi(Q_i)^2$.
- ϕ *cannot* increase the sparsity. Thus, $|\phi(Q_i)|_0 \le |Q_i|_0 \le \binom{kn+n/2}{n/2}$.
- Hence, $S_{\mathbb{F}}(f_d) \le s' \cdot \binom{kn+n/2}{n/2}$.

Proof of Theorem 1 (continued)

- We show that circuit-size$(P_{n,k}) = (kn)^{\omega(1)}$ (implying the family $\notin$ VP).
- Proof by contradiction. Suppose $P_{n,k}$ has a small-size circuit.
- SOS Decomposition shows that $P_{n,k}(\boldsymbol{y}) = \sum_{i=1}^{s'} c_i \cdot Q_i(\boldsymbol{y})^2$, where $\deg(Q_i) \le \deg(P_{n,k})/2 \le n/2$.
- Apply ϕ both side to get $f_d = \phi(P_{n,k}) = \sum_{i=1}^{s'} c_i \cdot \phi(Q_i)^2$.
- ϕ *cannot* increase the sparsity. Thus, $|\phi(Q_i)|_0 \le |Q_i|_0 \le \binom{kn+n/2}{n/2}$.
- Hence, $S_{\mathbb{F}}(f_d) \le s' \cdot \binom{kn+n/2}{n/2}$.
- Fix k, n appropriately and show:

$$\boxed{s' \le d^{o(\varepsilon)}, \text{ and } \binom{kn+n/2}{n/2} \le d^{1/2+\varepsilon/2}}.$$

Proof of Theorem 1 (continued)

- We show that circuit-size$(P_{n,k}) = (kn)^{\omega(1)}$ (implying the family $\notin \mathsf{VP}$).
- Proof by contradiction. Suppose $P_{n,k}$ has a small-size circuit.
- SOS Decomposition shows that $P_{n,k}(\boldsymbol{y}) = \sum_{i=1}^{s'} c_i \cdot Q_i(\boldsymbol{y})^2$, where $\deg(Q_i) \leq \deg(P_{n,k})/2 \leq n/2$.
- Apply ϕ both side to get $f_d = \phi(P_{n,k}) = \sum_{i=1}^{s'} c_i \cdot \phi(Q_i)^2$.
- ϕ *cannot* increase the sparsity. Thus, $|\phi(Q_i)|_0 \leq |Q_i|_0 \leq \binom{kn+n/2}{n/2}$.
- Hence, $S_{\mathbb{F}}(f_d) \leq s' \cdot \binom{kn+n/2}{n/2}$.
- Fix k, n appropriately and show:

$$\boxed{s' \leq d^{o(\varepsilon)}, \quad \text{and} \quad \binom{kn+n/2}{n/2} \leq d^{1/2+\varepsilon/2}}.$$

- Thus, $S_{\mathbb{F}}(f_d) \leq d^{o(\varepsilon)+1/2+\varepsilon/2} = o(d^{1/2+\varepsilon})$, a contradiction!

$\square$

Sum-of-cubes (SOC) model and Blackbox-PIT

Blackbox-PIT and Sum-of-cubes (SOC)

- Can SOS-hardness give PIT $\in \mathsf{P}$?

Blackbox-PIT and Sum-of-cubes (SOC)

- Can SOS-hardness give PIT $\in \mathsf{P}$? Ans: Don't know. Currently the best known is QP (when ε is constant), using result from [KI04].

Blackbox-PIT and Sum-of-cubes (SOC)

- Can SOS-hardness give PIT $\in$ P? Ans: Don't know. Currently the best known is QP (when ε is constant), using result from [KI04].
- Can we *strengthen* the condition/measure to put PIT $\in$ P?

Blackbox-PIT and Sum-of-cubes (SOC)

- Can SOS-hardness give PIT $\in \mathsf{P}$? Ans: Don't know. Currently the best known is QP (when ε is constant), using result from [KI04].
- Can we *strengthen* the condition/measure to put PIT $\in \mathsf{P}$? Ans: Yes!

Blackbox-PIT and Sum-of-cubes (SOC)

- Can SOS-hardness give PIT $\in \mathsf{P}$? Ans: Don't know. Currently the best known is QP (when ε is constant), using result from [KI04].
- Can we *strengthen* the condition/measure to put PIT $\in \mathsf{P}$? Ans: Yes!
- An n-variate polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ over a field $\mathbb{F}$ is computed as a *sum-of-cubes* (SOC) if

$$f(\boldsymbol{x}) \;=\; \sum_{i=1}^{s} c_i \cdot f_i(\boldsymbol{x})^3 \;, \tag{3}$$

  for some *top-fanin* s, where $f_i(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$.

Blackbox-PIT and Sum-of-cubes (SOC)

- Can SOS-hardness give PIT $\in \mathsf{P}$? Ans: Don't know. Currently the best known is QP (when ε is constant), using result from [KI04].
- Can we *strengthen* the condition/measure to put PIT $\in \mathsf{P}$? Ans: Yes!
- An n-variate polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ over a field $\mathbb{F}$ is computed as a *sum-of-cubes* (SOC) if

$$f(\boldsymbol{x}) = \sum_{i=1}^{s} c_i \cdot f_i(\boldsymbol{x})^3 , \tag{3}$$

  for some *top-fanin* s, where $f_i(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$.
    - Size of f in Eqn. (3) is no. of **distinct** monomials in f_i's i.e. $\left|\bigcup_{i=1}^{s} \mathrm{supp}(f_i)\right|$.

Blackbox-PIT and Sum-of-cubes (SOC)

- Can SOS-hardness give PIT $\in$ P? Ans: Don't know. Currently the best known is QP (when ε is constant), using result from [KI04].
- Can we *strengthen* the condition/measure to put PIT $\in$ P? Ans: Yes!
- An n-variate polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ over a field $\mathbb{F}$ is computed as a *sum-of-cubes* (SOC) if

$$f(\boldsymbol{x}) = \sum_{i=1}^{s} c_i \cdot f_i(\boldsymbol{x})^3 , \qquad (3)$$

  for some *top-fanin* s, where $f_i(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$.

  - Size of f in Eqn. (3) is no. of **distinct** monomials in f_i's i.e. $\left|\bigcup_{i=1}^{s} \mathrm{supp}(f_i)\right|$.

    Eg. $f(x) := x^3 + 6x^2 = (x+1)^3 - (x-1)^3 + x^3$. Size of f in this SOC representation is 2.

Blackbox-PIT and Sum-of-cubes (SOC)

- Can SOS-hardness give PIT $\in \mathsf{P}$? Ans: Don't know. Currently the best known is QP (when ε is constant), using result from [KI04].
- Can we *strengthen* the condition/measure to put PIT $\in \mathsf{P}$? Ans: Yes!
- An n-variate polynomial $f(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ over a field $\mathbb{F}$ is computed as a *sum-of-cubes* (SOC) if

$$f(\boldsymbol{x}) \;=\; \sum_{i=1}^{s} c_i \cdot f_i(\boldsymbol{x})^3 \;, \tag{3}$$

  for some *top-fanin* s, where $f_i(\boldsymbol{x}) \in \mathbb{F}[\boldsymbol{x}]$ and $c_i \in \mathbb{F}$.

  - Size of f in Eqn. (3) is no. of **distinct** monomials in f_i's i.e. $\left|\bigcup_{i=1}^{s} \mathrm{supp}(f_i)\right|$.

    Eg. $f(x) \;:=\; x^3 + 6x^2 \;=\; (x+1)^3 \;-\; (x-1)^3 \;+\; x^3$. Size of f in this SOC representation is 2.
  - Denote the *minimal size* by support-union $U_{\mathbb{F}}(f, s)$.

SOC-hardness : What to expect

- SOC is a *complete* model for $\mathrm{char}(\mathbb{F}) \neq 2, 3$ because for any $f(\boldsymbol{x})$:

$$f \;=\; (f+2)^3/24 \;+\; (f-2)^3/24 \;-\; f^3/12\;.$$

SOC-hardness : What to expect

- SOC is a *complete* model for $\mathrm{char}(\mathbb{F}) \neq 2, 3$ because for any $f(\boldsymbol{x})$:

$$f \;=\; (f+2)^3/24 \;+\; (f-2)^3/24 \;-\; f^3/12 \,.$$

- Trivially $U_{\mathbb{F}}(f, s) \leq |f|_0 + 1$, for any $s \geq 3$. By counting argument, $U_{\mathbb{F}}(f, s) \geq |f|_0^{1/3}$.

SOC-hardness : What to expect

- SOC is a *complete* model for $\mathrm{char}(\mathbb{F}) \neq 2, 3$ because for any $f(\boldsymbol{x})$:

$$f \;=\; (f+2)^3/24 \;+\; (f-2)^3/24 \;-\; f^3/12\;.$$

- Trivially $U_{\mathbb{F}}(f, s) \leq |f|_0 + 1$, for any $s \geq 3$. By counting argument, $U_{\mathbb{F}}(f, s) \geq |f|_0^{1/3}$.
- If $|f|_0 \approx d$, then $\Omega(d^{1/3}) \leq U_{\mathbb{F}}(f, s) \leq O(d)$.

SOC-hardness : What to expect

- SOC is a *complete* model for $\text{char}(\mathbb{F}) \neq 2, 3$ because for any $f(\boldsymbol{x})$:

$$f \;=\; (f+2)^3/24 \;+\; (f-2)^3/24 \;-\; f^3/12\,.$$

- Trivially $U_{\mathbb{F}}(f, s) \leq |f|_0 + 1$, for any $s \geq 3$. By counting argument, $U_{\mathbb{F}}(f, s) \geq |f|_0^{1/3}$.
- If $|f|_0 \approx d$, then $\Omega(d^{1/3}) \leq U_{\mathbb{F}}(f, s) \leq O(d)$.

Definition **(SOC-hardness).** A $\text{poly}(d)$-time explicit univariate polynomial family $(f_d)_d$, where f_d is of degree$-d$, is *SOC-hard*, if there exists a positive constant $\varepsilon' < 1/2$ such that $U_{\mathbb{F}}\left(f_d, d^{\varepsilon'}\right) = \Omega(d)$.

- SOC is a *complete* model for char($\mathbb{F}$) $\neq 2, 3$ because for any $f(\boldsymbol{x})$:

$$f \;=\; (f+2)^3/24 \;+\; (f-2)^3/24 \;-\; f^3/12\;.$$

- Trivially $U_{\mathbb{F}}(f, s) \leq |f|_0 + 1$, for any $s \geq 3$. By counting argument, $U_{\mathbb{F}}(f, s) \geq |f|_0^{1/3}$.
- If $|f|_0 \approx d$, then $\Omega(d^{1/3}) \leq U_{\mathbb{F}}(f, s) \leq O(d)$.

Definition **(SOC-hardness).** A $\mathrm{poly}(d)$-time explicit univariate polynomial family $(f_d)_d$, where f_d is of degree$-d$, is *SOC-hard*, if there exists a positive constant $\varepsilon' < 1/2$ such that $U_{\mathbb{F}}\left(f_d, d^{\varepsilon'}\right) = \Omega(d)$.

➢ Seems false over $\mathbb{F} = \mathbb{C}, \mathbb{R}$ [dimension argument].

SOC-hardness : What to expect

- SOC is a *complete* model for char($\mathbb{F}$) $\neq 2, 3$ because for any $f(\boldsymbol{x})$:

$$f \;=\; (f+2)^3/24 \;+\; (f-2)^3/24 \;-\; f^3/12\,.$$

- Trivially $U_{\mathbb{F}}(f, s) \leq |f|_0 + 1$, for any $s \geq 3$. By counting argument, $U_{\mathbb{F}}(f, s) \geq |f|_0^{1/3}$.
- If $|f|_0 \approx d$, then $\Omega(d^{1/3}) \leq U_{\mathbb{F}}(f, s) \leq O(d)$.

Definition **(SOC-hardness).** A $\mathrm{poly}(d)$-time explicit univariate polynomial family $(f_d)_d$, where f_d is of degree$-d$, is *SOC-hard*, if there exists a positive constant $\varepsilon' < 1/2$ such that $U_{\mathbb{F}}\left(f_d, d^{\varepsilon'}\right) = \Omega(d)$.

- ➢ Seems false over $\mathbb{F} = \mathbb{C}, \mathbb{R}$ [dimension argument].

$x^3 + y^3 = 1$ has *no* $\mathbb{Q}$ solution

- ➢ Instead fix $\mathbb{F} = \mathbb{Q}$, [*Natural* choice for PIT].

- SOC is a *complete* model for char($\mathbb{F}$) $\neq 2, 3$ because for any $f(\boldsymbol{x})$:

$$f \;=\; (f+2)^3/24 \;+\; (f-2)^3/24 \;-\; f^3/12\,.$$

- Trivially $U_{\mathbb{F}}(f, s) \leq |f|_0 + 1$, for any $s \geq 3$. By counting argument, $U_{\mathbb{F}}(f, s) \geq |f|_0^{1/3}$.
- If $|f|_0 \approx d$, then $\Omega(d^{1/3}) \leq U_{\mathbb{F}}(f, s) \leq O(d)$.

Definition **(SOC-hardness).** A $\text{poly}(d)$-time explicit univariate polynomial family $(f_d)_d$, where f_d is of degree$-d$, is *SOC-hard*, if there exists a positive constant $\varepsilon' < 1/2$ such that $U_{\mathbb{F}}\left(f_d, d^{\varepsilon'}\right) = \Omega(d)$.

- ➢ Seems false over $\mathbb{F} = \mathbb{C}, \mathbb{R}$ [dimension argument].

$x^3 + y^3 = 1$ has *no* $\mathbb{Q}$ solution

- ➢ Instead fix $\mathbb{F} = \mathbb{Q}$, [*Natural* choice for PIT].
- ➢ [Agrawal'20]: For $s = \Omega(d^{1/2})$, $U_{\mathbb{Q}}(f_d, s) = O(d^{1/2})$; for $s = \Omega(d^{2/3})$, $U_{\mathbb{Q}}(f_d, s) = \Theta(d^{1/3})$.

SOC-hardness : What to expect

- SOC is a *complete* model for char($\mathbb{F}$) $\neq 2, 3$ because for any $f(\boldsymbol{x})$:

$$f \;=\; (f+2)^3/24 \;+\; (f-2)^3/24 \;-\; f^3/12\,.$$

- Trivially $U_{\mathbb{F}}(f, s) \leq |f|_0 + 1$, for any $s \geq 3$. By counting argument, $U_{\mathbb{F}}(f, s) \geq |f|_0^{1/3}$.
- If $|f|_0 \approx d$, then $\Omega(d^{1/3}) \leq U_{\mathbb{F}}(f, s) \leq O(d)$.

Definition **(SOC-hardness).** A $\mathrm{poly}(d)$-time explicit univariate polynomial family $(f_d)_d$, where f_d is of degree$-d$, is *SOC-hard*, if there exists a positive constant $\varepsilon' < 1/2$ such that $U_{\mathbb{F}}\left(f_d, d^{\varepsilon'}\right) = \Omega(d)$.

- Seems false over $\mathbb{F} = \mathbb{C}, \mathbb{R}$ [dimension argument].
  $x^3 + y^3 = 1$ has *no* $\mathbb{Q}$ solution
- Instead fix $\mathbb{F} = \mathbb{Q}$, [*Natural* choice for PIT].
- [Agrawal'20]: For $s = \Omega(d^{1/2})$, $U_{\mathbb{Q}}(f_d, s) = O(d^{1/2})$; for $s = \Omega(d^{2/3})$, $U_{\mathbb{Q}}(f_d, s) = \Theta(d^{1/3})$.
- For $s < o(d^{1/2})$, we *conjecture* that *most* polynomials f_d are SOC-hard.

Theorem 2: SOC-hardness to PIT

Theorem 2: SOC-hardness to PIT

Theorem 2 (Efficient derandomization)

If there is an SOC-hard polynomial family, then blackbox-PIT $\in \mathsf{P}$.

Theorem 2: SOC-hardness to PIT

Theorem 2 (Efficient derandomization)

If there is an SOC-hard polynomial family, then blackbox-PIT $\in \mathsf{P}$.

Proof Idea. Assume f_d is SOC-hard for some ε'.

Theorem 2: SOC-hardness to PIT

Theorem 2 (Efficient derandomization)

If there is an SOC-hard polynomial family, then blackbox-PIT $\in \mathsf{P}$.

Proof Idea. Assume f_d is SOC-hard for some ε'.

- Convert it to $k = O(1)$-variate, ideg-n, $\mathsf{poly}(n^k)$-time-explicit polynomial $P_{n,k}$, using inverse-Kronecker map on f_d i.e. $P_{n,k}(x, x^{n+1}, \ldots, x^{(n+1)^{k-1}}) = f_d$.

Theorem 2: SOC-hardness to PIT

Theorem 2 (Efficient derandomization)

If there is an SOC-hard polynomial family, then blackbox-PIT $\in \mathsf{P}$.

Proof Idea. Assume f_d is SOC-hard for some ε'.

- Convert it to $k = O(1)$-variate, ideg-n, $\mathsf{poly}(n^k)$-time-explicit polynomial $P_{n,k}$, using inverse-Kronecker map on f_d i.e. $P_{n,k}(x, x^{n+1}, \ldots, x^{(n+1)^{k-1}}) = f_d$.
- Prove that $(P_{n,k})_n$ is a constant-variate circuit-*hard* family i.e. $\mathrm{size}(P_{n,k}) = n^{\Omega(1)}$. Then, use [Guo-Kumar-Saptharishi-Solomon'19] directly to conclude that PIT $\in \mathsf{P}$.

Theorem 2: SOC-hardness to PIT

Theorem 2 (Efficient derandomization)

If there is an SOC-hard polynomial family, then blackbox-PIT $\in \mathsf{P}$.

Proof Idea. Assume f_d is SOC-hard for some ε'.

- Convert it to $k = O(1)$-variate, ideg-n, $\mathrm{poly}(n^k)$-time-explicit polynomial $P_{n,k}$, using inverse-Kronecker map on f_d i.e. $P_{n,k}(x, x^{n+1}, \ldots, x^{(n+1)^{k-1}}) = f_d$.
- Prove that $(P_{n,k})_n$ is a constant-variate circuit-*hard* family i.e. $\mathrm{size}(P_{n,k}) = n^{\Omega(1)}$. Then, use [Guo-Kumar-Saptharishi-Solomon'19] directly to conclude that PIT $\in \mathsf{P}$.
- Proof by contradiction and use *useful* **SOC Decomposition**: Any polynomial f of degree d of circuit-size s can be written as $f = \sum_{i=1}^{\mathrm{poly}(s,d)} c_i Q_i^3$, where $\deg(Q_i) \leq 4d/11$. $[1/3 < 4/11 < 1/e]$

Theorem 2: SOC-hardness to PIT

Theorem 2 (Efficient derandomization)

If there is an SOC-hard polynomial family, then blackbox-PIT $\in \mathsf{P}$.

Proof Idea. Assume f_d is SOC-hard for some ε'.

- Convert it to $k = O(1)$-variate, ideg-n, $\mathsf{poly}(n^k)$-time-explicit polynomial $P_{n,k}$, using inverse-Kronecker map on f_d i.e. $P_{n,k}(x, x^{n+1}, \ldots, x^{(n+1)^{k-1}}) = f_d$.
- Prove that $(P_{n,k})_n$ is a constant-variate circuit-*hard* family i.e. size$(P_{n,k}) = n^{\Omega(1)}$. Then, use [Guo-Kumar-Saptharishi-Solomon'19] directly to conclude that PIT $\in \mathsf{P}$.
- Proof by contradiction and use *useful* **SOC Decomposition**: Any polynomial f of degree d of circuit-size s can be written as $f = \sum_{i=1}^{\mathsf{poly}(s,d)} c_i Q_i^3$, where $\deg(Q_i) \leq 4d/11$. $[1/3 < 4/11 < 1/e]$
- A binomial counting argument shows that small size of $P_{n,k}$ implies $U_{\mathbb{F}}(f_d, d^{\varepsilon'}) = o(d)$, a contradiction!

Conclusion

Conclusion

- Does the existence of a SOS-hard family solve PIT completely? The current proof technique *fails* to reduce from cubes to squares.

Conclusion

- ❑ Does the existence of a SOS-hard family solve PIT completely? The current proof technique *fails* to reduce from cubes to squares.
- ❑ Prove the existence of a SOS-hard family for the *sum of constantly* many squares.

Conclusion

- Does the existence of a SOS-hard family solve PIT completely? The current proof technique *fails* to reduce from cubes to squares.
- Prove the existence of a SOS-hard family for the *sum of constantly* many squares.
- Prove the existence of a SOC-hard family for a 'generic' polynomial f with rational coefficients ($\mathbb{Q}$).

Conclusion

- Does the existence of a SOS-hard family solve PIT completely? The current proof technique *fails* to reduce from cubes to squares.
- Prove the existence of a SOS-hard family for the *sum of constantly* many squares.
- Prove the existence of a SOC-hard family for a 'generic' polynomial f with rational coefficients ($\mathbb{Q}$).
- Can we optimize ε in the SOS-hardness condition and prove it for *any* $\omega(\sqrt{d})$?

Conclusion

- Does the existence of a SOS-hard family solve PIT completely? The current proof technique *fails* to reduce from cubes to squares.
- Prove the existence of a SOS-hard family for the *sum of constantly* many squares.
- Prove the existence of a SOC-hard family for a 'generic' polynomial f with rational coefficients ($\mathbb{Q}$).
- Can we optimize ε in the SOS-hardness condition and prove it for *any* $\omega(\sqrt{d})$? For eg: does proving an SOS lower-bound of $\sqrt{d} \cdot \mathsf{poly}(\log d)$, suffice to show $\mathsf{VP} \neq \mathsf{VNP}$?

Conclusion

- Does the existence of a SOS-hard family solve PIT completely? The current proof technique *fails* to reduce from cubes to squares.
- Prove the existence of a SOS-hard family for the *sum of constantly* many squares.
- Prove the existence of a SOC-hard family for a 'generic' polynomial f with rational coefficients ($\mathbb{Q}$).
- Can we optimize ε in the SOS-hardness condition and prove it for *any* $\omega(\sqrt{d})$? For eg: does proving an SOS lower-bound of $\sqrt{d} \cdot \mathsf{poly}(\log d)$, suffice to show $\mathsf{VP} \neq \mathsf{VNP}$?